RTI/7473/029-02S November 2001

RADAR BASELINE TURBULENCE DETECTION ALGORITHM IMPLEMENTATION FOR PC:

DESCRIPTION AND CHANGES

and

USER'S GUIDE

TECHNICAL REPORT

NASA Contract NAS1-99074

Task Assignment 1029

Prepared for

National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

3040 Cornwallis Road • Post Office Box 12194 • Research Triangle Park, North Carolina 27709-2194 USA

RTI/7473/029-2S November 2001

RADAR BASELINE TURBULENCE DETECTION ALGORITHM IMPLEMENTATION FOR PC:

DESCRIPTION AND CHANGES

and

USER'S GUIDE

By
Joseph H. White
Charles L. Britt, Ph.D.

Center for Aerospace Technology
RTI

TECHNICAL REPORT

NASA Contract NAS1-99074

Task Assignment 1029

Prepared for

National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

TABLE OF CONTENTS

1.0 Introduction 1

2.0 Development of Baseline Turbulence Detection Algorithm 2

2.1 Introduction 2
2.2 Approach 2
2.3 Radar Measurements 3
2.4 Radar Parameters 5
2.5 Corrections to the Total Wind Variance Estimation 5
2.6 Implementation of Algorithms 7
2.7 Algorithm Testing with Simulated and Flight Data 8
2.8 Summary 8

3.0 A Post Processing Version of the Real-time Baseline Software For Turbulence Detection 9

3.1 Description 9
3.2 Installation 11
3.3 Operation 12

4.0 References 16

APPENDIX A A-1

APPENDIX B B-1

1.0 Introduction

An algorithm was developed for the detection of atmospheric turbulence using sampled return data from experimental weather radar. The initial development of the algorithm was initially done in Fortran under the Radar Algorithm Visualization program (RAVE) using simulated data and then refined using data collected during early turbulence data gathering flights. Concurrent with this development was the development of a real-time processing display system for use on the 757 aircraft to support both turbulence and Enhanced Vision System experiments. This real-time system was centered on a high-performance dual-processor personal computer with multiple digital signal processors (DSP) to handle the quantity of data produced by the radar. For turbulence detection, code from the RAVE system was ported to "C" for execution primarily on the DSPs. Recently, as the RAVE Fortran code was upgraded based on experience with actual flight data, those enhancements were ported to the real-time system to improve its detection capability.

The software running on the real-time system and its laboratory-based counterpart used for software development represents a useful tool for examination of recorded data. However, since the software required DSPs to operate, its use was limited to these two systems. To expand the usability of that software, the DSP code was modified to run on the standard Pentium processor and to accept recorded data from a file. The result is a standalone program that can process and display recorded data using the Baseline algorithm. An alternate version was also created by the same technique using the NCAR Efficient Spectral Processing Algorithm (NESPA).

This report describes the Baseline algorithm and the standalone implementation of it. Section 2 describes the algorithm in detail including the radar detection algorithm and the estimation of aircraft accelerations based on the sensed level of turbulence. A memo listing spcific changes made to the initial implementation to produce the version described here is included in Appendix A. Section 3 describes the software implementation of the baseline for use on the standalone processor called PostProcB. "PostProc" refers to post processing version and the "B" reflects the use of the Baseline algorithm in that version. Appendix B describes a variation of this software called PostProcNESPA in which the NESPA processing code from NCAR replaces the Baseline implementation.

2.0 Development of Baseline Turbulence Detection Algorithm

2.1 Introduction

This section outlines the current baseline algorithms for implementing an aircraft turbulence hazard detection capability for the Collins Research Data Radar under development for use in the NASA B757 for turbulence experiments. The algorithms discussed in this memo will be improved upon and enhanced as additional experience is realized and other algorithms are developed and tested. The performance of the baseline algorithms has been assessed using flight test data from NASA B757 flights during the year 2000.

2.2 Approach

The general approach to the development of the baseline algorithms can be summarized as follows. It is assumed that sufficient radar reflectivity is present to accurately measure winds along the projected aircraft path. Two radar measurements are used to infer the severity of the turbulent (fluctuating or gust) component of the wind field along the aircraft path. These are 1) the total variance of the Line-Of-Sight (LOS) wind gust component at each radar range bin, and 2) the correlation distance of the LOS mean wind fluctuations along the flight path. The first measurement permits estimation of the variance of the wind gust components and the latter measurement gives an estimate of the scale length of the turbulent wind field and can also be used to estimate other statistical parameters of the field (Ref. 1). These measurements are then used in algorithms (developed by AeroTech, Inc) that relate the gust estimates to the standard deviation of "g" loads on the aircraft. These g-load estimates then provide a prediction of the expected severity of the turbulence along the flight path of the aircraft.

The assumption is made that the random turbulent gust field is isotropic. This assumption implies that the gust component variances in the vertical and transverse directions are directly related to the variance of the LOS component. Thus, the variance of the LOS component measured along the LOS by the radar also provides an estimate of the vertical and transverse component variances along the LOS. Although all gust components have an effect on aircraft motion, for stochastic turbulence it is the vertical component that is primarily responsible for uncomfortable and potentially hazardous motion due to large aircraft pitch angle and altitude changes (pitch and plunge mode - see Ref. 2).

To relate the radar measurements of total variance and correlation distance to the aircraft motion, an algorithm developed by Dr. Roland Bowles of AeroTech, Inc. is used. This algorithm uses a set of hazard tables for each type of aircraft and requires inputs providing aircraft altitude, true airspeed, and weight, as well as the radar measurements. From these inputs, the algorithm estimates the "g" load standard deviation that would be experienced by the aircraft if it were flying through the area where the measurements were taken.

Thus, the radar-estimated g-force standard deviation along the aircraft path provide a predictive measure of the turbulence hazard level that will be encountered as the predicted path is flown, assuming conditions do not change rapidly along the path. Thresholds and displays can be defined to trigger warnings to the aircrew that potentially hazardous turbulence may be encountered within X seconds unless the projected aircraft path is modified.

2.3 Radar Measurements

Measurements - As mentioned above, two basic radar measurements are used to characterize the random turbulent wind field. These measurements are taken while the radar is in a horizontal (azimuth) scanning mode. The measurements are 1) the total variance of the longitudinal component of the gust field and 2) the correlation distance or scale size of the turbulence along the radar LOS. The total variance is estimated for each radar range bin, whereas the correlation distance is estimated for each azimuth line. Various averaging schemes (over time or space) may be used to smooth the measurements.

A major problem exists if the weather radar reflectivity is not sufficient to provide good estimates of these parameters and means are provided in the algorithms to use signal-strength thresholds to suppress erroneous estimates. In environments with small volumes of reflectivity, it is especially difficult to use bin-to-bin variations to estimate the gust correlation distance. If sufficient valid measurements are not available for correlation estimates, a fixed default value is used.

Total variance of gust velocity components - The total variance of the Doppler velocity at a particular range bin along the radar LOS can be found from a combination of the ensemble averages of the square of the Doppler spectral width and the variance of the mean velocity in the bin. According to Ref. 3, for the usual range of weather radar parameters, this calculation of total variance should be independent of the radar pulse width and antenna beam shape. In equation form:

$$\sigma_p^2 = < \sigma_v^2 > + \sigma_m^2$$

$$where: \quad < \sigma_v^2 > = average\ of\ \ square\ of\ \ Doppler\ width$$

Eq. 1

$$\sigma_m^2 = variance\ of\ \ mean\ bin\ velocity$$

$$\sigma_p^2 = total\ variance\ of\ \ LOS\ bin\ velocity$$

The time period or spatial volume over which the averages should be taken is an input parameter with the best value to be determined. The averaging must be sufficient to estimate the statistics of the LOS turbulent gust component, but not so extensive that it increases the gust variance due to slower changing (large scale-low frequency) velocity trends in the ambient wind field that have little effect on the turbulence hazard to the aircraft. Since the gust field is assumed to be isotropic, the variance of the LOS velocity also provides an estimate of the variance of the vertical component of gust velocity. Therefore, $\sigma_w^2 \approx \sigma_p^2$ where the subscript *w* refers to the vertical gust velocity component. This variance is used in the g-loading algorithms to determine the effect of the gust statistics on the aircraft motion.

In order to provide an estimate of the portion of the total gust variance at each range bin due to bin-to-bin variations, a running spatial average or equivalent spatial low-pass filter implementation is used. To calculate this bin-to-bin component at a given range bin, the bin velocities over a radial/transverse area surrounding the bin of interest are used in the variance calculation. Time averages are used when aircraft/antenna scan motions permit multiple looks at the same volume of space.

Turbulence scale size - The scale size or correlation distance (L) of the longitudinal component of velocity is calculated from the variations of the measured velocities in each range bin along a radar LOS. Each set of radar, measured velocities vs. range (along the lines-of-sight near the projected path of the aircraft) is considered a sample function. The average value taken over all range bins is first subtracted from the sample function (i.e., velocity values are averaged over a specified range and this average is subtracted from each velocity value) to obtain a zero mean sample function. Next, the normalized autocorrelation function of each sample function is calculated. A set of autocorrelation functions taken over a specified (input parameter) angular scan range near the aircraft path is then averaged. A measure of the gust correlation distance is then found by integrating the resulting correlation function over range and then dividing the value of the integral by the total range extent (Ref. 2). This distance is assumed to represent the scale size of the LOS turbulence component along the aircraft path for this angular region of the radar scan. The same calculated scale size is used for g-load calculation at all ranges along a particular azimuth line.

The smallest turbulence scale size that can be measured by the radar is determined by the width of the radar pulse. The largest size is limited by the maximum radar range. For civil aircraft gust load testing, a value for L of 2500 ft (762 m) has been accepted as representative (Ref. 2). In implementing the algorithm, the L calculation is tested for validity to assure that it is within a range of reasonable values (300 to 2000 m is suggested). If, for some reason (such as a lack of sufficient reflectivity) a valid value cannot be determined, a value of 762 m is used as a default.

2.4 Radar Parameters

Collins Research R/T - The research radar used in the NASA B757 for turbulence hazard detection and other research is user programmable over a large range of parameters. These are listed as follows:

Frequency	9318 to 9382 MHz
Pulse Width	1 to 24 microseconds
Pulse Repetition Frequency (PRF)	500 Hz to 14.4 kHz
Number of Pulses	1 to 255
Number of Range Bins	1 to 256
Range Bin Width	1 to 51 microsecs
Azimuth Center	-85 to 85 degrees
Azimuth Range	5 to 180 degrees
Scan Rate	45 or 22.5 deg/sec
Number of Bars	1 to 15
Tilt Spacing	0.0 to 5.0 degrees
Display Output	5 to 320 nm range
Control Input	RS-232
Transmitted Power	180 watts

The various parameters listed above can be set to specific values using the RS232 radar control input and software furnished by Rockwell/Collins.

2.5 Corrections to the Total Wind Variance Estimation

Error Sources - Several factors other than the turbulent wind component can affect the magnitude of the spectral width component of the total variance and should be considered as error sources. These factors are pointed out by Doviak and Zrnic (Ref. 3) and include transverse shear due to the mean wind, radial wind shear and antenna scan effects. Shear in this context means linear variations in the strength of the Doppler component of the wind over the dimensions of the radar beam/pulse volume that persist over a time that is long compared to the radar measurement interval. Equations are provided in (Ref. 4) for calculation of these effects. If the radar can measure the wind shears, these error sources can be removed to obtain a corrected variance due to the turbulent wind component only.

The severity of these error sources was investigated in (Ref. 7). For the radar parameters anticipated, the errors due to antenna scanning and radial wind shear were found to be small.

The errors caused by transverse and vertical wind shear are strongly dependent on the magnitude of the shear and the radar range. For a range of 10 km, the error in the spectral width calculation is approximately 4.8 m/s for a shear of 30 m/s per km. At a range of 6 km, the error for the same shear has dropped below 3 m/s.

From the above considerations, it can be concluded that, for the radar parameters selected for the research radar, the vertical and transverse mean wind shear are potentially important error sources in the estimation of spectral width. If these error sources are not removed, the shear terms could cause false turbulence indications in a strong shear environment.

Removal of Error Sources - Using the equations provided in Ref. 3 for the magnitude of the shear error as a function of the radar parameters and the magnitude of the mean shear, the errors could theoretically be removed. In order to do this, the mean shear of the Doppler component must be estimated by the radar in the radial, vertical, and horizontal directions. Estimation of the shear in the radial and horizontal directions can be estimated with the radar in an azimuth scan mode, whereas estimation of vertical shear requires scanning in a vertical plane and may not be feasible because of scan time limitations and the desire for a reasonably fast update rate on turbulence measurements. The basic equations and calculation techniques for correcting the total gust variance for the shear error terms discussed above are given in Ref. 1.

Algorithms for g-loading - The algorithm developed by R. Bowles (Ref. 6) was implemented in the software to predict the inertial load forces in "g" units (g-loading) that may be experienced by the aircraft during flight. The routine requires five inputs: the velocity of the aircraft in m/s; the aircraft altitude in meters; an estimate of aircraft weight in kLbs; estimated correlation length in meters; and an estimate of the variance of the wind in m/s. It is assumed that the turbulence scale length will be between 200 and 3000 meters. In the event that turbulence scale length falls outside those parameters or is unavailable, a turbulence scale length of 500 meters is assumed.

The routine uses the inputs to calculate a normalized velocity V_n based on aircraft altitude A according to the following equation:

$$V_n(A) = 492.64 + (6.2023 * A) + (0.092343 * A * A) \quad \text{ft/s} \qquad \text{Eq. 2}$$

This equation assumes the aircraft altitude is between 10 kft and 30 kft. For altitudes of less than 10 kft a value of $V_n = 761.89$ ft/s is used, while for altitudes over 30 kft the input velocity is used after being converted to the proper units. The routine then calculates a scaling factor (SF) based on this equation:

$$SF = (a - (b * \log10(L) * (180.0/\text{weight}) (\text{velocity}/V_n)) \qquad \text{Eq. 3}$$

where:

L = correlation length in meters
weight = weight of the aircraft in kLbs
V_n = normalized velocity (m/s) calculated in previous step
a, b = coefficients (see below)

The routine uses the aircraft altitude as an index into lookup tables for the coefficients a and b and estimates appropriate values using linear interpolation.

Finally the variance of the wind is multiplied by the factor SF to arrive at a value for the variance of g-loading in units of g. This is the value returned by the routine to the following algorithms for further processing and display.

2.6 Implementation of Algorithms

Selection of a Baseline Set - For initial implementation of the algorithms that are planned for simulations, post-flight data analysis, and real-time use in the NASA research radar system, it was desirable to avoid undue complexity in the algorithm set. This was due to time and funding limitations and the necessity to meet the overall project schedule and milestones, including scheduled installation of the research radar in the NASA B757 aircraft and associated flight tests. For the initial testing of the baseline algorithms in the NASA experimental radar, corrections for steady wind shear were not implemented. It is anticipated that shear corrections may be incorporated in future real-time and post-flight data processing if the wind shears encountered in flight data cause errors in measurements.

The baseline set of algorithms selected for implementation in the real-time and post-processing systems include algorithms for the following:

- Filtering anomalous impulses in the (time domain) I and Q signals
- Weighting the I and Q signals prior to Fourier transforming with either Hann or Hamming weights
- Performing a complex Fast Fourier Transform (FFT) up to 512 points
- Scaling I & Q values to weather reflectivity (dBZ) for display
- Changing the Doppler spectral reference (zero Doppler) to remove the effect of aircraft ground speed
- Providing an adaptive power threshold for spectral moment estimation
- Calculating spectral moments (spectral width and velocity) using a spectral averaging technique that is not sensitive to velocity fold-over
- Estimating the turbulence standard deviation due to bin-to-bin velocity variations
- Estimating the gust correlation length over radial lines of sight
- Estimating the total gust variance of the radial component of velocity at each range cell over the radar scan area
- Using total gust variance and correlation length to estimate and display the predicted g-loading on the aircraft (for the B757)
- Averaging velocities, spectral widths, and g-loading over ”n” azimuth lines and “m” range bins to provide spatial smoothing

The algorithms for estimation of g-loading from spectral widths, gust correlation distance, and aircraft parameters were furnished by Dr. R. Bowles, as previously mentioned.

2.7 Algorithm Testing with Simulated and Flight Data

Simulation Runs – Simulated weather fields generated by NCAR were used by ADWRS to generate I and Q data sets over several simulated radar scans for use in a Radar Algorithm Visualization program (RAVE). This program implements the baseline algorithms and provides radar maps for examination at various stages in the estimation of the total variance of the wind and the g-loading. The effects of shear corrections can also be evaluated. An advantage of the simulated data is that the statistics of the (simulated) radar measurements can be compared with the known statistics of the input weather field.

Numerous simulations were conducted with various weather fields and radar parameters. Results of these simulations are given in Ref. 7.

Flight Data Runs - The RAVE1 program was modified to read and process radar I & Q data and ARINC 429 aircraft bus data from flight tests conducted near Greeley, CO, in June, 1999 and flight test data from the NASA B757 aircraft taken during the fall of 2000. The results of applying the baseline algorithms to these data are given in Ref. 7 and were presented to the turbulence team at the PDT meeting on April 24, 2001.

2.8 Summary

A baseline set of turbulence detection algorithms for airborne weather radar has been developed and implemented in computer codes designed to assess the performance of the algorithms. The algorithms were exercised using simulated radar data, from the Greeley, CO, flight tests and from the NASA research radar on the B757. The baseline algorithms are intended to be a starting point in the development of a set of reliable turbulence detection algorithms that are suitable for operational use.

Assessment of the performance of the algorithms has been done using tools to read and plot flight test, simulated data, and in situ acceleration data. These tools include the ADWRS algorithm development (RAVE) programs discussed previously, as well as utility programs for reducing the large (20 Hz sampling) aircraft data bus files to smaller files with slower sampling rates, and MATLAB programs (Ref. 5) for viewing and plotting aircraft and weather data files.

3.0 A Post Processing Version of the Real-time Baseline Software For Turbulence Detection

3.1 Description

History - A real-time processing and display system was developed for the experimental weather radar used on the NASA 757 for the Turbulence Detection and Mitigation Element of the Aviation Safety program. This system consisted of an industrial dual-Pentium computer with four TI 'C6701 DSPs. The software package for this system consisted of three parts:

1. DSP-based software for the calculation of spectra and moments, as well as associated processing.
2. Pentium-based software for system control and data movement.
3. Pentium-based software for the generation of map and stripchart displays from data passed through shared memory.

The first version of this system was used on flights in September to December 2000. Based on analysis of the data collected during those flights, modifications were made to the baseline algorithm to improve its detection accuracy and those improvements are presently being integrated in the real-time system.

Software for the flight system was developed on a second system identical with that used on the aircraft with the exception of additional DSPs used to generate a simulated radar data stream from data read from a file. Because of its capability of processing recorded data, the development system became a tool for reviewing recorded data. However, the software was restricted to use on systems with Spectrum Daytona DSP boards installed and configured identically to the development system, thus limiting the use of this software to a single station.

Porting of the DSP code to the Pentium processor, along with additional user interface changes would produce a software package for data review and analysis that could be run on ordinary PCs. This would allow review of data by persons who do not have access to the development system. This effort was further justified in the fall of 2001 by a failure in all three DSP boards. This made them unavailable for integration of Baseline upgrades, but a port to the Pentium processor allowed continuation of the development work while the boards were being repaired.

Architecture - The post-processing version of the real-time software consists of two separate executables: A display program called radar_ui and a processing program called PostProcB.

The program radar_UI is a LabCVI-based program that receives its input via shared memory and displays the results on up to 4 ppi displays and a strip chart (graph) display. An additional spectral map display is provided but is not used for this application. Radarui.exe is a version of radar_UI.exe that is configured for running on processor 0x0001 of a dual processor system running under Windows NT.

Configuration files are read by radar_UI.exe from the config subdirectory, located under the working directory for radar_UI. Important files in this directory are:

colors.cfg — A text file of RGB values for 256 colors used for false color plots in PPI displays. Each color is specified by three integers (for red, green and blue) each having a value between 0 and 255. The file has 256 lines, one for each color, with the three color values on each line separated by commas. Color 0 represents the lowest amplitude value and Color 253 represents the highest. Color 254 is displayed when the display value cannot be computed (i.e., in the case of insufficient signal). Color 255 is reserved (Note: the use of 256 colors provides for a color scale that will provide smooth transitions throughout the range displayed. However, groups of identical RGB values may be defined in the color.cfg file to provide a smaller number of discrete color levels so that contour lines of the plotted parameters are more discernable. For the real-time system, 16 colors are used with each color defined in 16 entries in the table, except for the last color, which occupies only 14 entries due to the dedicated use of the top two entries of the table.

default.cfg — Window status (open/closed), size, and position are saved in this file for restoration the next time the program is opened.

The PostProcB executable contains radar data reading and processing code. It is based on modified code from three sources:

1. File reading code from the program used to generate a simulated data stream (RecSimulator.c in the development system)
2. Processing code from the DSP software
3. Control and display interface software from the host portion of the real-time software.

All code was combined in a single file except for DSP code which, when possible, was left as separate files to enable porting of the modified code back to the real time system. The source files comprising the PostProcB project are as follows:

PostProcB.c — Main program. PostProcB.c is responsible for reading the *.iq file, converting header and i/q data to appropriate formats, performing spectral calculations, calling external processing routines, interfacing with user, and passing data to display. This routine includes code for reading the input file from RecSimulator.c, code from the input DSP processor on the real time system for spectral computations, and host code from the real time system for display interface.

dsp0b_b.c — Code from second of three DSPs in real time system. Code in this file is responsible for averaging the spectra from adjacent range bins and shifting the spectra to remove the aircraft velocity doppler component.

dsp1a_b.c	Code from third of three DSPs in real-time system. Code in this file is responsible for calculation of moments and bin-to-bin variance.
commonb.h	Fixed parameter and structure definitions used in compilation of PostProcB.exe.
sharemem.h, map_def.h	Structures used in shared memory for communication with display software.

Other files used by PostProcB.exe at run time are:

RecentFiles.txt	List of files recently accessed by PostProcB. File may be edited with text editor.
Rtdisp.cfg	Configuration parameters – primarily ranges of displayed parameters.

3.2 Installation

Installation of the software consists only of running an install program for the radar_ui and copying files from the installation CD to specified directories.

Display software may be installed by running the setup.exe located in the radar_UI_install directory of the installation CD. This will create a radar_UI directory a specified location containing the files several files including the following:

Radar_UI.exe
config subdirectory containing:
    colors.cfg
    default.cfg

Note that two executables are included in this directory. Radar_UI.exe, the larger file, is intended for Win9x or single processor NT systems. The other executable contains the SetProcessorAffinity() function to control which processor it runs on, and is limited to NT. A shortcut to the executable may be added to increase ease of use. Alternative .cfg files used for the real-time system are included in the radar_UI/config directory on the installation CD.

Installation of PostProcB software for processing consists of creating a directory, C:\PostProcB and copying the following files into it and clearing their "read only" attribute:

PostProcB.exe
RTdisp.cfg
RecentFiles.txt (optional. If not present, new file will be created).

For convenience, PostProcB.exe may be associated with files with extension ".iq" to enable files to be processed by simply double-clicking on them in Windows Explorer.

PostProcB may be installed in another directory if one of the two actions is taken:

1. Set the environment variable "PostProcB" to the directory where the PostProcB software is installed.
2. Recompile PostProceB.exe with the arguments to _chdir() changed to the proper directory.

If program modifications are desired, and Visual C++ 6.0 is available, a suitable development environment may be set up by copying the following source files into the PostProcB directory:

PostProcB.c
dsp0b_b.c
dsp1a_b.c
common.h
sharemem.h
map_def.h

The following project files should be copied into the same directory:

PostProcB.dsp, .dsw, .ncb, .opt, and .plg.

Upon compilation, the object files and updated executable will be placed in Debug directory under the directory containing the source. The previous version of executable copied from the installation CD may be deleted. Although the VC++ project files were not set up specifically to exclude relocation to directories other than C:/PostProcB, it is possible that the project may not compile properly if located in another directory. If such behavior is observed, a quick solution is to delete the project files, create a new, empty console application, and add the source files to it.

3.3 Operation

To view recorded turbulence radar data, it is necessary only to start radar_ui and PostProcB by any convenient means, including double clicking on the executable in Windows Explorer, typing in the executable name in a command window, or clicking on an associated file (recommended for PostProcB only).

At least a control panel will be opened by Radar_ui. Additional windows that were open when Radar_ui was closed will be reopened. Additional windows may be opened by clicking the various buttons on the control panel. Windows that are not desired may be closed with the standard "x" box in the upper right corner of the window border. Configurations defined by number of windows open, their size and placement may be assigned a file name and stored for later restoration using the "save" and "open" buttons on the control window.

If PostProcB.exe is initiated by double clicking on an associated file, it will begin processing data from the file with no further action required. If it is initiated via shortcut or double clicking on the PostProcB.exe in Explorer or under MyComputer, the user will be presented with a list of files that were recently accessed and prompted to select an entry from the list or enter a new file name. Up to 16 file names are maintained in the list of recently accessed files. Opening an additional file after the list of 16 is full will result in the deletion of the top entry, shifting all entries up one slot, and adding the new entry at the bottom of the list. Note that the entry is added irrespective of whether the file is found or not. The list of entries is contained in the file, RecentFiles.txt, which may be edited as desired with a standard text editor to remove files not found or add new file names.

Once the file is opened and data is being displayed, the user may control data flow via the keyboard using the following case-insensitive commands listed in Table 1. Additional program parameters may be changed by accessing the menu ('M' command in Table 1). Commands available under this menu are described in Table 2.

TABLE 1. KEYBOARD COMMANDS ACTIVE WHEN PostProcB.exe IS RUNNING

Command	Action
S	Enter (or remain in) **S**ingle step mode and display the next frame (dwell) of data.
B	Enter (or remain in) single step mode and display the previous frame (dwell) of data **(B**ackup**).** (**Note**: If averaging over multiple azimuth lines is enabled, then the display results will reflect the average of 3 consecutive frames of data retrieved. This is intended to reflect the average of data from three azimuth lines centered on the azimuth displayed in the text window. This will only be true for sequences where three or more dwells are retrieved in the same direction, either moving forward through data or stepping backward)
J	**J**ump to arbitrary position if file specified in bytes. User is prompted to specify whether a jump is desired to a relative displacement from the present position or to a specific location relative to the beginning of file. (File position is displayed continuously in the console window for reference.)
M	Bring up **M**enu for modification of program processing parameters. Options under this menu are defined in Table 2.
X	E**X**it the program.
Any other key	Exit single step mode.

TABLE 2. PROGRAM PARAMETERS THAT MAY BE CHANGED VIA PROGRAM PARAMETER MENU.

Command	Action
T	Toggle time-domain editor ON/OFF. Time-domain editor detects and smoothes outliers in i/q values for successive pulses within a range bin to remove interference from other radars.
W	Toggle windowing used prior to using FFT for spectral estimation. If ON, a Hann window (raised cosine) window is applied with compensation for reduction in energy that would occur with application of the window.
S	Toggle spectral averaging ON/OFF. Spectral averaging is the process of averaging spectra from two adjacent range bins within an azimuth line and placing the result in the more distant bin.
M	Toggle range/azimuth moment averaging ON/OFF. If ON, moments and loads from a region 3 azimuth lines wide by 5 range bins are averaged and the result displayed at the center of the region, raw (unaveraged) data is displayed in the first and last two range bins. Only averaged azimuth lines are displayed.
P	Toggle between CFAR processing and thresholding based on a fixed received power level. In CFAR processing, signal and noise are calculated by sorting spectra in order of magnitude and computing the average of the lowest 25% of the spectral components to compute the noise, and the average of the remainder to compute the signal. If the signal exceeds the noise by a specified factor (see "C" below), moments and loads are computed. Otherwise no data is displayed for that bin. If CFAR processing is off, moments are calculated anytime the received power exceeds a specified value (see "R" below).
Z	Toggle the display in the first window between received power (dBm) and reflectivity (dBZ).
D	Specify the width of each azimuth line drawn to the map displays. Default value is 2.5 degrees so that azimuth lines with 2 degree spacing will overlap slightly.
R	Set the received power threshold used when CFAR processing is disabled (see "P" above).
C	Set signal/noise ratio required for moments and loads to be calculated when CFAR processing is enabled (see "P" above). The ratio is specified as a dimensionless number, i.e., not in dB.
B	Specify range bin for which computed values are displayed in text (console) window.
X or <RET>	Exit the menu dialog.

4.0 References

1. Panofsky, H. A. and John A. Dutton, Atmospheric Turbulence, John Wiley & Sons, New York, 1984.

2. Hoblit, Frederic M., Gust Loads on Aircraft: Concepts and Applications, AIAA Education Series, AIAA Inc., Washington, DC, 1988.

3. Doviak, R. J. and D. S. Zrnic, Doppler Radar and Weather Observations, Academic Press, Inc., 1984.

4. Britt, C. L., Users Guide for an Airborne Windshear Doppler Radar Simulation (ADWRS) Program (Version 2.0), RTI Report 3042-68-1, June 1990.

5. Kelly, C. W., Software Elements for Independent Testing of Turbulence Models and Algorithms, RTI Report 7473/013-01S, March 2000.

6. Bowles, R. L, Hazard Estimation & Implementation Guidelines, Slides presented at the meeting of the Turbulence Detection Team, Hampton, Virginia, June, 2000.

7. Britt, C. L., Turbulence Hazard Detection: Development and Initial Assessment of Baseline Radar Algorithms, RTI report 7473/013-05S, November, 2000.

APPENDIX A

Technical Memo:
Changes to Baseline Turbulence Detection Algorithms
(made for Version 2)

RTI Technical Memo
C.L.Britt
April 12, 2001

1.0 Changes to Baseline Turbulence Detection Algorithms

The following lists the major changes to the baseline algorithms incorporated in the algorithm evaluation programs since the December flights.

1. The method of calculating spectral width was changed from a frequency-domain pulse-pair technique to a spectral averaging technique. This was done because with large headwinds or tailwinds (>15 m/s) and with the current radar PRF of 3000 pps, the weather spectra will fold over at the Nyquist frequency and the frequency-domain pulse-pair can product erroneous results.

A Fortran subroutine is available now.

2. The time-domain impulse filter furnished by NASA was reworked to provide better results.

A Fortran subroutine is available now.

3. Provision was made to average spectra over two adjacent range bins if desired.

(Option) This averages spectral lines in bin n and bin n-1 prior to taking moments to improve the SNR.

4. Provision was made to average moments and g-loading over N range bins and L azimuth lines as specified by an operator input. The averaged values are then displayed.

(Option) Have used N=5 and L=3 with good results (must be odd numbers).

5. The g-load table values for given altitude and weight were uniformly increased by a factor of 1.09 in accordance with instructions from R. Bowles.

Trivial change.

6. Techniques for calculating bin-to-bin velocity standard deviations and along-bin correlations were modified to better

handle missing data points (i.e., bins with signal level below threshold).

Default values were used if less than X consecutive values were available (above threshold) for calculation.

7. The thresholding method was changed so that the moment calculations use a CFAR (constant false alarm rate) threshold calculated from the spectra, whereas the reflectivity (dBZ) displays use a fixed SNR threshold. Required threshold constants are supplied by the operator.

The reflectivity threshold is set so received power must be greater than –108 dBm (Operator input) for the reflectivy value to be displayed, otherwise a value of light grey is displayed.

The CFAR moment threshold for a calculation is based on the average power of the smallest 25% of spectral lines in the bin(s). For a moment or g-load to be calculated and displayed, the total power must be greater than the power in the smallest 25% of lines by a factor of X dB (Operator input - usually 8 dB). If below this threshold, a light grey is displayed.

NOTE: Currently, a different (better) method of doing this threshold is being examined.

NOTE: We are also going through the code to make sure that other radar modes will work okay. For example, modes with longer pulses or higher pulse rates. Also, items like gate_delay and transmit pulse length are being taken from the input data file instead of being hard coded as may have been done originally since we were in a hurry.

Another change is a revision in the distance to the start of first range bin to the value given by Collins at the last meeting. This was fixed at 150m and is now calculated from the gate delay.

```
BINSZ=(FLTDAT.BINWIDTH)*0.2E-6*CC/2.       From Jonathan Lai
RMINRAD=(NCDAT.IGATEDEL)*BINSZ  !5/10/01
```

APPENDIX B

A Post Processing Version of the Real-time NESPA Software For Turbulence Detection

Introduction

The PostProcB standalone code for implementation of the Baseline algorithm was modified to produce a version containing the NESPA algorithm being developed by National Center for Atmospheric Research (NCAR). The new program is called PostProcNESPA. Since both PostProcB and PostProcNESPA are derived from the real-time software used with the radar on the NASA 757, they share many common elements. Thus the job of producing a standalone NESPA implementation became a task of replacing baseline specific code in PostProcB with NESPA code from NCAR and NESPA specific interface routines from the real-time system.

Because of the common heritage of the NESPA implementation with the baseline, much of the history, architecture, and installation is the same for both implementations. This Appendix addresses details of the NESPA implementation where it differs from the baseline description in the main body of this report. A description of the basic NESPA processing is not included since that code was not developed at RTI.

Architecture

PostProcNESPA uses similar architecture as the baseline implementation except that the main loop that reads and processes data is more complex. Code in this loop must detect which bar is being read, and store data in arrays based on antenna position, i.e., data is stored beginning with the first dwell of a three-bar set. Also the spectra for 5 azimuth lines are averaged which involves delaying the header information associated with each dwell so that the information for the center line of each set of averaged azimuth lines is associated with the averaged data.

Source files for the PostProcNESPA implementation are as follows:

PostProcNESPA.c	Main program. PostProcNESPA.c reads and decodes the *.iq file, computes spectra, calls routine to remove aircraft velocity doppler component, performs spectral averaging across 5 azimuth lines, applies 5x5 median filter to azimuth vs. range spectra, sorts data and places it in arrays for NESPA processing, calls NESPA routines, and passes data to display routines via shared memory.
Dsp0b_b.c	Code from second of three DSPs in real time system. Code in this file is responsible for shifting the spectra to remove the aircraft component and center the spectra so that zero velocity is in the mid-point of the spectral array.
CommonNespa.h	Parameter and structure definitions used in the compilation of PostProcNESPA.exe

Configuration files and shared memory header files are similar in function to those for the baseline implementation.

NESPA-specific routines provided by NCAR and stored in the Nespa subdirectory under the PostProcNESPA directory include the following:

cfrd_func.c
check_time.c
checktime.h
confread.c
confread.h
cramers_rule.c
cramers_rule.h
get_line.c
hazard_load_table.hdr
hazard_load_table.tbl
hazard_m2_table.hdr
hazard_m2_table.tbl
init_nespa_hazard.c
init_nespa_hazard.h
make_dist_wt.c
make_dist_wt.h
ncar_hazard.c
ncar_hazard.h
nespa.conf
nespa_confidence.c
nespa_confidence.h
nespa_config.c
nespa_config.h
nespa_hazard.c
nespa_hazard.h
nespa_pass1.c
nespa_pass1.h
nespa_pass2.c
nespa_pass2.h
nespa_passes.c
nespa_passes.h
nr.h
NRsort.c
read_hazard_table.c
read_hazard_table.h
vssver.scc
wp_const.h

Installation

Installation of PostProcNESPA is the same as installation of PostProcB with the exception that it is intended to be copied to the subdirectory C:/PostProcNESPA. The PostProcNESPA subdirectory may be located other than at the root, C:/, if an environment variable PostProcNESPA is set to the actual subdirectory or if the subdirectory passed as an argument to _chdir() in the code is changed to reflect the actual location.

The radar_ui software installed for use with the baseline will work with the NESPA implementation with no modification.

Operation

Operation of PostProcNESPA is similar to the baseline implementation except for the specific commands available under the menu command, M. These are described in Table A-1.

TABLE A-1. PROGRAM PARAMETERS THAT MAY BE CHANGED VIA PROGRAM PARAMETER MENU.

Command	Action
T	Toggle time-domain editor ON/OFF. Time-domain editor detects and smoothes outliers in i/q values for successive pulses within a range bin to remove interference from other radars.
W	Toggle windowing used prior to using FFT for spectral estimation ON/OFF. If ON, a Hann window (raised cosine) window is applied with compensation for reduction in energy that would occur with application of the window.
S	Toggle spectral averaging ON/OFF. Spectral averaging is the process of averaging spectra from two adjacent range bins within an azimuth line and placing the result in the more distant bin.
M	Toggle median filter (5 range bins by 5 spectral bin) ON/OFF
D	Specify the width of each azimuth line drawn to the map displays. Default value is 2.5 degrees so that azimuth lines with 2 degree spacing will overlap slightly.
A	Enter value to be used for aircraft weight in pounds to be used with NESPA processing.
B	Specify range bin for which computed values are displayed in text (console) window.
X or <RET>	Exit the menu dialog.